Hindawi
Discrete Dynamics in Nature and Society
Volume 2020, Article ID 9475782, 11 pages
https://doi.org/10.1155/2020/9475782

*Research Article*

# Fuzzy Matching Template Attacks on Multivariate Cryptography: A Case Study

**Weijian Li**,[1] **Xian Huang**,[1] **Huimin Zhao**,[1] **Guoliang Xie**,[2] **and Fuxiang Lu**[1]

[1]*School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou 510665, China*
[2]*Dept of Electronic and Electrical Engineering, University of Strathclyde, Glasgow G11XQ, UK*

Correspondence should be addressed to Huimin Zhao; zhaohuimin@gpnu.edu.cn

Received 19 April 2020; Accepted 25 May 2020; Published 20 June 2020

Guest Editor: Jianbiao Zhang



Multivariate cryptography is one of the most promising candidates for post-quantum cryptography. Applying machine learning techniques in this paper, we experimentally investigate the side-channel security of the multivariate cryptosystems, which seriously threatens the hardware implementations of cryptographic systems. Generally, registers are required to store values of monomials and polynomials during the encryption of multivariate cryptosystems. Based on maximum-likelihood and fuzzy matching techniques, we propose a template-based least-square technique to efficiently exploit the side-channel leakage of registers. Using QUAD for a case study, which is a typical multivariate cryptosystem with provable security, we perform our attack against both serial and parallel QUAD implementations on field programmable gate array (FPGA). Experimental results show that our attacks on both serial and parallel implementations require only about 30 and 150 power traces, respectively, to successfully reveal the secret key with a success rate close to 100%. Finally, efficient and low-cost strategies are proposed to resist side-channel attacks.

## 1. Introduction

With the upcoming quantum computers, traditional cryptosystems face huge challenges. Public-key cryptosystems such as Rivest–Shamir–Adleman (RSA) and elliptic curve cryptography (ECC), whose security relies on the difficulty of certain number theoretic problems, are under great threat of quantum attack. In as early as 1994, Peter Shor proposed an algorithm on a quantum computer that efficiently solved such number theoretic problems in polynomial time. Afterward, Monz et al. [1] presented the realization of a scalable Shor algorithm in 2016, which means that once large-scale quantum computers appear, public-key cryptosystems will become insecure. Meanwhile, for symmetric-key primitives, larger keys are required to resist the quantum attack to some extent.

Since Shor's discovery, the theory of post-quantum cryptography has developed significantly. Many cryptographic schemes proposed in the literature, such as code-based cryptography [2] and lattice-based cryptography [3], show great potentiality to resist quantum attacks, while multivariate cryptography is one of the most promising candidates [4]. Afterward, numerous cryptosystems based on multivariate quadratic polynomials have been proposed, such as unbalanced oil-and-vinegar (UOV) and its variant [5], Rainbow [5, 6], ZHFE [7], and CHNN-MVC [8].

At Eurocrypt 2006, Berbain et al. [9] presented the first multivariate stream cipher scheme denoted as QUAD, which is referred to as a practical and provable secure stream cipher, as well as a pseudorandom number generator (PRNG). In 2009, Berbain et al. [10] revisited the stream cipher of QUAD and proposed the provable security arguments supporting its conjectured strength for suitable parameter values. The provable security of QUAD relies on the hardness of solving systems of multivariate quadratic equations. Bardet et al. [11] presented a cryptanalysis algorithm with a complexity bounded by $O(2^{134.56})$, which means this cryptanalysis method cannot put into practice.

In recent years, GPUs are widely used in cloud computing and blockchain, which faces huge security challenges

to guarantee data security and user privacy [12–14]. Several GPU acceleration schemes for multivariate systems are proposed to make it suitable for security of cloud computing and blockchain in the quantum world [15, 16]. In 2014, Tanaka et al. [15] proposed two efficient parallelization algorithms and a GPU-based multivariable quadratic polynomial system. Furthermore, they proposed several effective parallel implementations of QUAD on GPU to accelerate the computing of quadratic polynomials. In 2018, Liao et al. [16] proposed a GPU acceleration framework for high-order multivariate cryptography systems, where the GPU acceleration schemes made multivariate cryptosystems feasible for cloud computing and blockchain.

Moreover, multivariate cryptosystems are in general computationally efficient, which supports the use of the Internet of Things (IoT) devices. IoT is essentially a network of pervasive devices such as RFID tags, sensors, ASICs, and smart cards, which have rigid cost constraints in terms of area, memory, computing power, and battery supply. Traditional cryptosystems are not entirely applicable to the IoT devices since they are too expensive for such pervasive devices. At fast software encryption (FSE) 2010, Billet et al. [17] showed that QUAD can be converted to efficiently construct a privacy-preserving authentication protocol for RFID with provable security. Arditti et al. [18] presented a QUAD implementation and regarded it as the smallest provably secure stream cipher so far. The smallest QUAD implementation requires only 2961 GE, which makes it a competitive candidate for IoT security. Also, Hamlet et al. [19] proposed a throughput-optimized parallel implementation of QUAD for more secure application scenarios in 2015.

The implementation of cryptography needs to take a wide range of physical attacks into account, especially side-channel attacks and fault attacks. Side-channel attacks exploit the dependency between physical information (e.g., power consumption, electromagnetic leaks, and timing information) and secret key to enable a divide-and-conquer attack to reveal the key part by part. Typical side-channel attacks include nonprofiled attacks (e.g., correlation power analysis (CPA) [20], mutual information analysis (MIA) [21]) and profiled attacks (e.g., template attacks (TA) [22–28] and other machine learning-based side-channel attacks [28–34]). Profiled side-channel attacks are the most powerful attacks, which received a lot of attention in recent years. Samples of power traces are regarded as features, and feature selection methods are needed to reduce the computational complexity and increase the prediction accuracy [28]. Afterward, machine leaning techniques including maximum-likelihood strategy [22–28], SVM [28–30], random forest (RF) [28, 29], $k$-nearest neighbors (KNN) [31], neural networks (NNs) [32], and deep learning (DL) [33, 34] are widely applied to build the prediction model. Profiled side-channel attacks include a profiling/training phase and a matching/predicting phase. In the profiling/training phase, machine learning algorithms are fed with labelled power traces captured from a reference device to build the prediction model. In the matching/predicting phase, prediction models are used to predict the correct labels for those power traces captured from a target device.

Template attack was first proposed at CHES'02 [22], which efficiently revealed the key by a maximum-likelihood strategy, and was rapidly accepted as the strongest form of side-channel attack. Original template attack matched only a single power trace, which sometimes failed in the practical attack. Agrawal et al. [23] proposed template-based DPA attack to accumulate the matching results of power traces, which significantly improved the success rate. Özgen et al. [24] combined classification algorithms with template attacks in the matching phase to improve the efficiency of attacks. Choudary and Kuhn [25] tackled some of the practical obstacles of template attacks, such as pooled covariance matrices, compression methods, and incompatibility of templates across different devices. Zhang [27] theoretically analyzed the exact relationship between the success rate of template attack and values of different parameters, including signal-to-noise, number of interesting points, and number of power traces. From the viewpoint of machine learning, Picek et al. [28] adopted feature selection techniques to improve the attack efficiency. They concluded that L1 regularization wrapper and linear SVM hybrid methods performed consistently well for all data sets.

Although side-channel attacks have been developed over 20 years, research about side-channel attacks on multivariate cryptosystems is still in the early stages. Several literatures about side-channel attack on multivariate cryptosystems were published. In as early as 2005, Okeya et al. [35] analyzed the power leakage of addition operations modulo $2^{32}$ of SHA-1 and successfully recovered the secret information of SFLASH, which is the first successful power analysis attack on multivariate cryptography in practice. Later, in 2013, Hashimoto et al. [36] proposed a theoretical method based on fault attack to reveal the partial key of MPKC systems. Yi and Li [37] proposed a fault attack and DPA on ASIC implementation of enTTS scheme in 2017. In 2018, Park et al. [38] presented a correlation power analysis attack against the Rainbow and UOV schemes on an 8-bit AVR microcontroller that yields full secret key recoveries. In 2019, based on the work of Hashimoto et al., Krämer and Loiero [39] complemented the research on fault attacks of multivariate signature schemes. However, their attacks do not lead to complete key recovery on Rainbow and UOV. Recently, Li et al. [40] proposed a CPA attack against serial implementation of QUAD on FPGA. Their work efficiently revealed the secret key but still requires further work to improve success rate.

Li et al. proposed the practical CPA cryptanalysis on serial QUAD (2, 160, 160) with a much lower complexity, but the success rate is only around 85%. Because of the low signal-to-noise ratio, classic template attack and template-attack DPA attack cannot exactly match the templates to achieve a satisfactory success rate. To tackle this issue, we have proposed template-based least-square power analysis on serial QUAD (2, 160, 160). The main contributions of our paper can be highlighted as follows:

(1) By applying the least-square technique to enable fuzzy matching of the templates, which can find the best matching via minimizing the squared sum of errors. As a result, the proposed practical can achieve a success rate of nearly 100%.

(2) We also extend the template-based least-square power analysis attack to explore the leakage of parallel implementation of QUAD (2, 160, 160), which has successfully and efficiently revealed the secret key with a success rate also close to 100%.

(3) For multivariate cryptography, all monomials and polynomials can be computed in an arbitrary order to break the link between the power consumption and the secret key. We propose two low-cost hiding countermeasures for serial and parallel implementations, respectively, which show great potential to resist side-channel attacks.

The remaining paper is organized as follows: in Section 2, we review the mathematical definition, serial and parallel FPGA implementations of the QUAD stream cipher; in Section 3, the template-based least-square power analysis attacks on the serial and parallel FPGA implementation of the QUAD are presented; experimental results of our attacks are given in Section 4; efficient and low-cost countermeasures to resist side-channel attacks are discussed in Section 5; and Section 6 concludes the paper.

## 2. Preliminaries

*2.1. Multivariate Cryptography.* Generally, the mathematical definition of a multivariate quadratic equation with $n$ variables over GF ($q$) can be written as follows:

$$Q(x_1, \ldots, x_n) = \sum_{1 \le i \le j \le n} \alpha_{ij} x_i x_j + \sum_{1 \le i \le n} \beta_i x_i + \gamma, \quad (1)$$

where $\alpha_{ij}$, $\beta_i$, and $\gamma$ are all coefficients over GF ($q$). Note that the degree of polynomial is up to 2; otherwise, new variables will be introduced to keep the polynomial of degree 2. A multivariate quadratic system $Q(X)$ consisting of $m$ multivariate quadratic equations in $n$ variables over GF ($q$) is defined as

$$Q(X) = \{Q_1(X), \ldots, Q_m(X)\}, \quad X = x_1, \ldots, x_n. \quad (2)$$

Given a multivariate quadratic system $Q(X)$, the MQ problem is defined as to find a value $X = x_1, \ldots, x_n$, if any, such that $Q_i(X) = 0$ for all $1 \le i \le m$. The MQ problem is proved to be NP hard, even in the smallest finite filed GF (2) [10].

A particular QUAD stream cipher in $n$ variables over GF ($q$) is specified as QUAD ($q$, $n$, $r$), which computes $n + r$ polynomials per round. As shown in Figure 1, QUAD ($q$, $n$, $r$) consists of an output function $S_{\text{out}}(X) = (Q_{n+1}(X), \ldots, Q_{n+r}(X))$ to produce $r$ outputs as the keystream, and an update function $S_{\text{in}}(X) = (Q(X)1, \ldots, Q_n(X))$ is used to generate $n$ outputs to update $X$ for the next round. The parameters $q$, $n$, and $r$ and the coefficients $\alpha_{ij}$, $\beta_i$, and $\gamma$ for $S_{\text{in}}$ and $S_{\text{out}}$ are public.

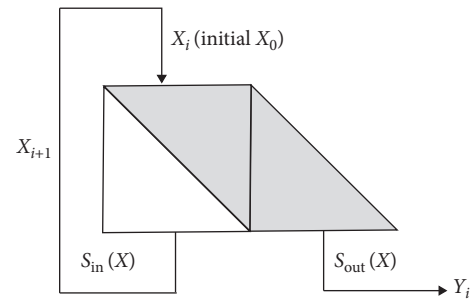


Figure 1: Stream cipher generation in QUAD [9].

The QUAD cipher expands a secret initial state $X_0 \in \text{GF}(q)^n$ into a sequence of secret states $X_0, X_1, X_2, \ldots \in \text{GF}(q)^n$ and a sequence of output vectors $Y_0, Y_1, Y_2, \ldots \in \text{GF}(q)^r$.

QUAD (2, 160, 160) is a practical version with the security level of at least $2^{80}$, which is strongly recommended in [10]. QUAD (2, 160, 160) has 160 variables over GF (2), which outputs 160 bits per round, resulting in a set of 320 multivariate quadratic equations.

From a perspective of implementation, operations over GF (2) are more efficient than those over larger fields. Moreover, the monomial forms $x_i \cdot x_i$ and $x_i$ are equal over GF (2); therefore, $\alpha_{ij} x_i x_j$ and $\beta_i x_i$ can be computed together. In the case of randomly generated $\alpha_{ij}$ and $\gamma$, equations of QUAD over GF (2) can be simplified as

$$Q(X) = \sum_{1 \le i \le j \le n} \alpha_{ij} x_i x_j + \gamma, \quad (3)$$

which brings great benefits in terms of efficiency and security.

*2.2. FPGA Serial Implementation of QUAD.* Arditti et al. [18] proposed a compact serial implementation of QUAD, which is believed to be the smallest provably secure stream cipher. As shown in Figure 2, the implementation consists of two main components. The first one is a nonlinear feedback shift register (NFSR), in which the coefficients of $\alpha$ and $\gamma$ are randomly generated. Each monomial of the equation is computed by the second component at every clock tick and accumulated to a result register. Multivariate quadratic equations $Q_1(X)$, $Q_2(X)$, ..., $Q_{n+r}(X)$ are computed sequentially. At every clock tick, the NFSR generates the coefficient. Once a new monomial $\alpha_{ij} x_i x_j$ of polynomial $Q_k(X)$ is computed, its contribution will be accumulated to the temporary register $Q_k$. After $n(n+1)/2 + 1$ clock cycles, the polynomial $Q_k(X)$ is computed, and the above process is repeated for $Q_{k+1}(X)$.

*2.3. FPGA Parallel Implementation of QUAD.* Hamlet and Brocato [19] presented two throughput-optimized parallel implementations of QUAD for a much higher throughput. A QUAD (2, 128, 128) version with the security level of approximately $2^{64}$ is considered, which can be easily extended to another version in GF (2) such as QUAD (2, 160, 160). The coefficients $\alpha$ and $\gamma$ are randomly generated and stored in ROM. Multivariate quadratic equations

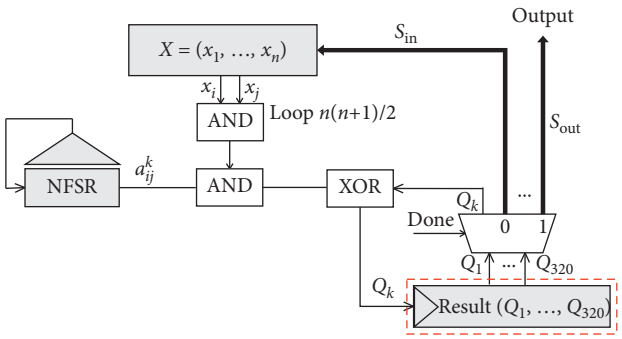


Figure 2: Serial implementation of QUAD (2, 160, 160) [18].

$Q_1(X), Q_2(X), \ldots, Q_{n+r}(X)$ are still computed sequentially, while $n$ monomials of polynomial $Q_k(X)$ are computed in parallel at a time to achieve a higher throughput.

As shown in Figure 3, the FPGA parallel implementation of QUAD (2, 160, 160) is summarized as follows:

(1) Sequentially compute the equations $Q_1(X)$, $Q_2(X)$, ..., $Q_{320}(X)$ by step (2) to step (6).

(2) Initialize the internal state $X$ and rotated $X$ with the secret key and the initialization vector.

(3) Calculate $\eta_i = x_i$ AND rotated $x_i$, $1 \le i \le 160$ simultaneously.

(4) Load 160-bit coefficients $\alpha$ from ROM, feed $\eta$ and $\alpha$ into 8-input AND-XOR modules to compute $M_c = \sum_{3 \ge i \ge 0} \alpha_{4c-i} \eta_{4c-i}$, $1 \le c \le 40$, and store $M_c$ in temporary register $P_c$.

(5) Calculate $V_{c_1} = \sum_{3 \ge i \ge 0} M_{4c_1-i}$, $1 \le c_1 \le 10$, by XOR modules. Compute $Q_k(X) = Q_k(X) \oplus \sum_{1 \le i \le 10} V_i$ and store the value in result register $Q$.

(6) Rotate the internal state rotated $X$ by one bit, and go to step 3 to compute next 160 monomials until all monomials of polynomial $Q_k(X)$ are completed, which requires $\lceil (n+1)/2 \rceil = 81$ loops. Note that, in the last loop, only half of the above modules are enabled to compute the last 80 monomials.

(7) Repeat the above steps until all quadratic equations are computed.

## 3. Proposed Attack on Implementation of QUAD

*3.1. Power Leakage Model.* A typical CMOS transistor consumes dynamic power when its output signal is converted. Figure 4(a) shows the changing process of a register when the output signal is converted from 0 to 1. A charging current from the power supply to the output capacitance $C_L$ and a transient short-circuit current from CMOS transistor are generated. On the contrary, Figure 4(b) shows the discharging process when the output signal is converted from 1 to 0. Only the instantaneous short-circuit current is generated through CMOS transistor.

As a result, conversions of the output signal are focused since dynamic power is the major power consumption of the digital logical circuits of ASIC and FPGA. Denote the power consumption of CMOS transistor by $P_{ij}$ when its signal converts from $i$ to $j$, where $i$ and $j$ equal to 0 or 1. $P_{01}$ and $P_{10}$ consume dynamic power, while $P_{00}$ and $P_{11}$ consume only static power. As a result, it generally holds that $P_{00} \approx P_{11} \ll P_{01}, P_{10}$.

Therefore, the power consumption when writing data to a register depends on the number of bit-flips. A hamming distance (HD) model well summarizes the power consumption of a register transition from a previous state to a new state.

Regarding multivariate cryptosystems, which consist of a large number of monomials and polynomials, registers are indeed required to store monomial and polynomial values during the encryption. Serial implementation, for instance, monomials are computed sequentially and accumulated to the temporary register $Q_k$, as identified by rectangle in Figure 2.

The value of register $Q_k$ changes to $Q_k \oplus \alpha_{ij} x_i x_j$ for all monomials. The power consumption of register $Q_k$ can be concluded as follows:

$$L(Q(x)) = HD\left(Q_k, Q_k \oplus \alpha_{ij} x_i x_j\right) = HW\left(\alpha_{ij} x_i x_j\right), \quad 1 \le k \le 160. \tag{4}$$

Consequently, an attacker is possible to predict secret keys $x_i$ and $x_j$ by observing the power consumption of registers $Q_l$.

Other than serial implementations, parallel implementations compute 160 monomials simultaneously. 4 monomials are accumulated by an AND-XOR module and stored into temporary register $P_c$. According to the parallel implementation described in Section 2.3, when computing the first 160 monomials, the values $M_c$, $1 \le c \le 40$, stored into the temporary register $P_c$ are

$$M_c = a_{4c-3,4c-3} x_{4c-3} x_{4c-3} \oplus a_{4c-2,4c-2} x_{4c-2} x_{4c-2} \oplus a_{4c-1,4c-1} x_{4c-1} x_{4c-1} \oplus a_{4c,4c} x_{4c} x_{4c}, \quad 1 \le c \le 40. \tag{5}$$

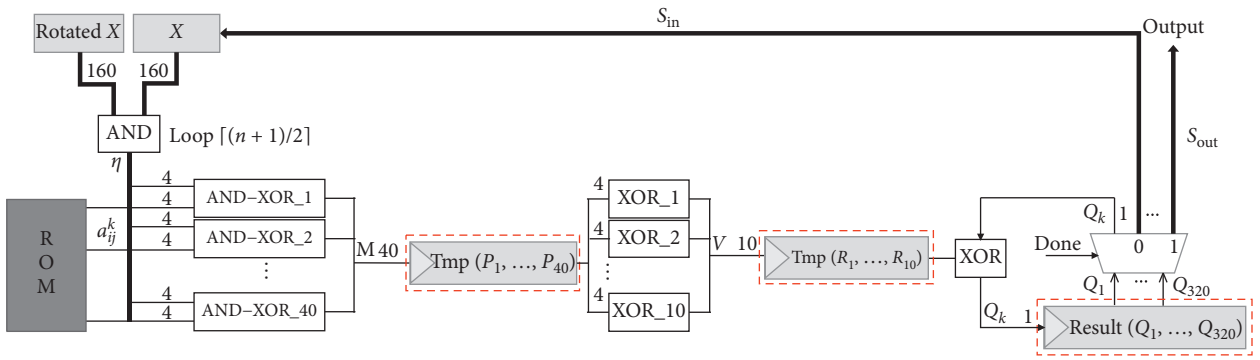


Figure 3: Parallel implementation of QUAD (2, 160, 160) [19].

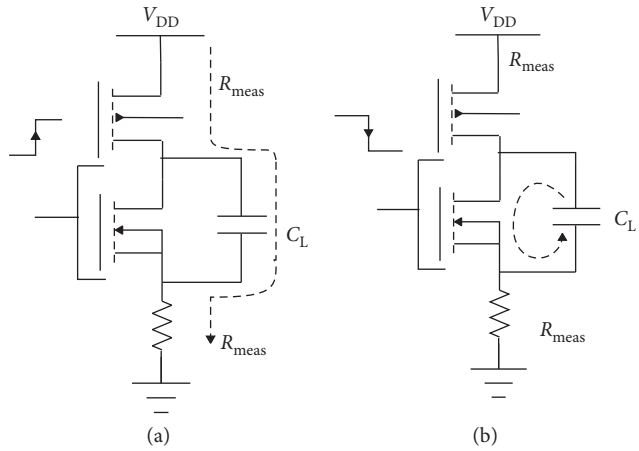


Figure 4: Charging and discharging process of register.

After rotating the internal state rotated $x$ by one bit to compute the next 160 monomials, the values $M_c', 1 \le c \le 40$, stored into registers $P_c$ are

$$\begin{aligned} M_c' &= a_{4c-3,4c-2}x_{4c-3}x_{4c-2} \oplus a_{4c-2,4c-1}x_{4c-2}x_{4c-1} \oplus a_{4c-1,4c}x_{4c-1}x_{4c} \oplus a_{4c,4c+1}x_{4c}x_{4c+1}, \quad 1 \le c \le 39, \\ M_c' &= a_{157,158}x_{157}x_{158} \oplus a_{158,159}x_{158}x_{159} \oplus a_{159,160}x_{159}x_{160} \oplus a_{1,160}x_{160}x_1, c = 40. \end{aligned} \tag{6}$$

Therefore, the values of registers $P_c, 1 \le c \le 40$, change from $M_c$ to $M_c'$, and the power leakage model of parallel implementations can be defined as

$$L(Q(x)) = HD(M_c, M_c') = HW(M_c \oplus M_c'), \quad 1 \le c \le 40. \tag{7}$$

*3.2. Template-Based Least-Square Power Analysis Attack.* Classic side-channel attacks, such as DPA, CPA, and MIA, require a large number of power traces to reveal the key, which means that different plaintexts are needed to be encrypted with the same key for obtaining as much power traces as possible. However, multivariate cryptosystems usually contain limited quadratic equations. Take QUAD as an example, and the key of QUAD is constantly updated after each round of encryption, which only generates $n + r$ power traces with the same key. In this case, machine learning-based side-channel attacks such as template attack have inherent advantages, which can extract the key with much fewer target power traces.

Machine learning-based side-channel attacks are the most powerful attacks. Based on a maximum-likelihood strategy, template attacks reveal the secret key efficiently, which consist of a profiling phase and a matching phase. Classic template attacks match only a single power trace and reveal the key by the Bayes theorem in the matching phase. However, there is not enough valuable information in a

single power trace to reveal the correct key in practical situation; hence, the classic template attack has little prospect of success rate.

To solve this problem, template-based DPA attack was proposed as follows [23]:

$$p\left(k_j \mid T\right) = \frac{\left(\prod_{1\le i\le D} p\left(t_i \mid k_j\right)\right)\cdot p\left(k_j\right)}{\sum_{1\le l\le K}\left(\left(\prod_{1\le i\le D} p\left(t_i \mid k_l\right)\right)\cdot p\left(k_l\right)\right)}, \tag{8}$$

which accumulates the matching degree of each power trace during template matching to improve the success rate.

Unfortunately, due to the low signal-to-noise ratio, the accurate matching method of template-based DPA attack is not applicable. For this reason, we proposed a template-based least-square (LSQ) power analysis attack, which reveals the key by fuzzy matching. As described in Figure 5, the main idea of template-based LSQ is as follows:

(1) Choose a strategy to build templates: according to the power leakage models in equations (4) and (7), two templates need to be built, corresponding to leakage values 0 and 1.

(2) Collect power traces to build templates: two groups of power traces are collected according to different leakage values.

(3) Select interesting points (features): samples of power traces are regarded as features, which include relevant, irrelevant, and redundant features. Feature selection methods are needed to select the most relevant features to improve the attack efficiency. Feature selection methods [28] such as squared pairwise T-differences (SOST), Pearson correlation, principal component analysis (PCA), linear SVM wrapper, and L1 regularization are investigated. In our experiments, the Pearson correlation method is chosen to search interesting points, which can lead to excellent classification performance. 25 interesting points for serial implementation and 35 interesting points for parallel implementation with the highest $\rho_{t,hw}$ are selected, where $\rho_{t,hw}$ is defined as

$$\rho_{t,hw} = \frac{\operatorname{cov}(t, hw)}{\sigma_t \sigma_{hw}}. \tag{9}$$

(4) Build templates with interesting points: two templates $h_i = (m_i, C_i)$ corresponding to leakage values 0 and 1 are built, respectively, by covariance matrix $C_i$ and mean vector $m_i$, where $m_i$ and $C_i$ are defined as

$$m_i = \frac{1}{d}\sum_{1\le j\le d} t_j^i,$$
$$C_i(u,v) = \frac{1}{d-1}\sum_{1\le l\le d}\left(N_{ul} - \overline{N}_u\right)\cdot\left(N_{vl} - \overline{N}_v\right), \tag{10}$$

(5) Match templates: power traces $T = \{t_1, t_2, \ldots, t_D\}$ with the same key are captured from the device under attack to match the templates, respectively. Template that leads to the highest probability $p(t_j; (m_i, C_i))$ indicates the correct leakage value, where $p(t_j; (m_i, C_i))$ is defined as

$$p\left(t_j; (m_i, C_i)\right) = \frac{\exp\left(-(1/2)\cdot\left(t_j - m_i\right)'\cdot C_i^{-1}\cdot\left(t_j - m_i\right)\right)}{\sqrt{(2\cdot\pi)^T\cdot\det\left(C_i\right)}}. \tag{11}$$

Denote such leakage values corresponding to $T = \{t_1, t_2, \ldots, t_D\}$ as $H = [h_1, h_2, \ldots, h_D]$.

(6) Reveal the correct key. We map the hypothetical intermediate values into leakage values by equation (7) and compare with $H = [h_1, h_2, \ldots, h_D]$ to reveal the correct key. Taking attack on parallel implementations, for instance, denote the hypothetical leakage values by $S = \{s_1, s_2, \ldots, s_{32}\}$, where $s_i = \{s_{i1}, s_{i2}, \ldots, s_{iD}\}$. The least-square method defined as

$$F(i) = \sum_{1\le j\le D}\left(s_{ij} - h_j\right)^2, \quad 1\le i\le 32, 1\le j\le D, \tag{12}$$

is applied to compare the hypothetical leakage values with the leakage values revealed by the template attack, where $i$ is the key hypothesis. Finally, the correct key is revealed by

$$\text{key} = \arg\min F(i). \tag{13}$$

## 4. Experimental Results and Discussion

As shown in Figure 6, our experimental setup includes a standard evaluation board SAKURA-G, an oscilloscope, and a computer. SAKURA-G is designed for hardware security, which equips with two separate Spartan-6 FPGA chips. One chip serves as the control chip, while another serves as the cryptographic chip. Cryptographic chip performs encryption operations, while the control chip controls the data flow and communicates with the oscilloscope and computer. During encryption, power consumptions of the cryptographic chip are measured by the oscilloscope which is triggered by the control chip. Finally, power analysis attacks are performed on the computer.

We first perform a side-channel attack on serial implementation of QUAD (2, 160, 160). In the template building phase, 3000 power traces with different keys and coefficients are captured from a reference device, based on which 25 interesting points are selected by the CPA peak method. Next, we collect two groups of power traces corresponding to leakage values 0 and 1 in equation (4), and each group consists of 25 power traces. Finally, we build two templates, and the result is shown in Figure 7.

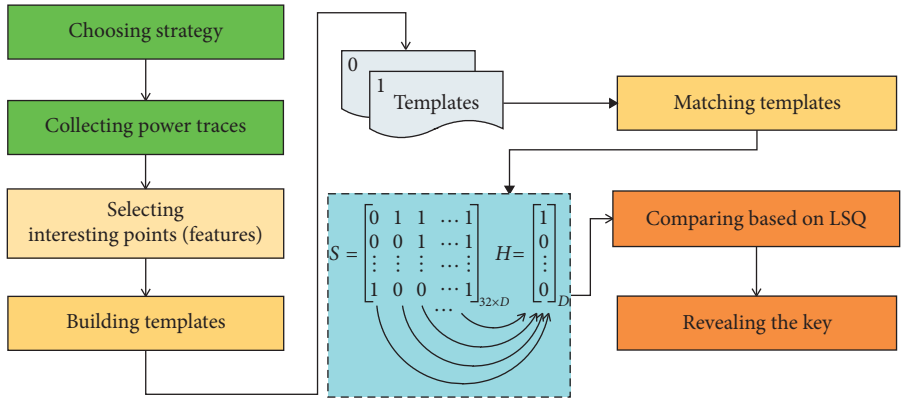


Figure 5: The flow of template-based LSQ attack.

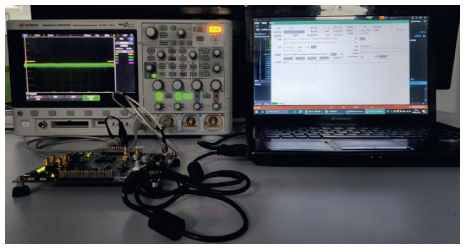

Figure 6: Experimental setup.

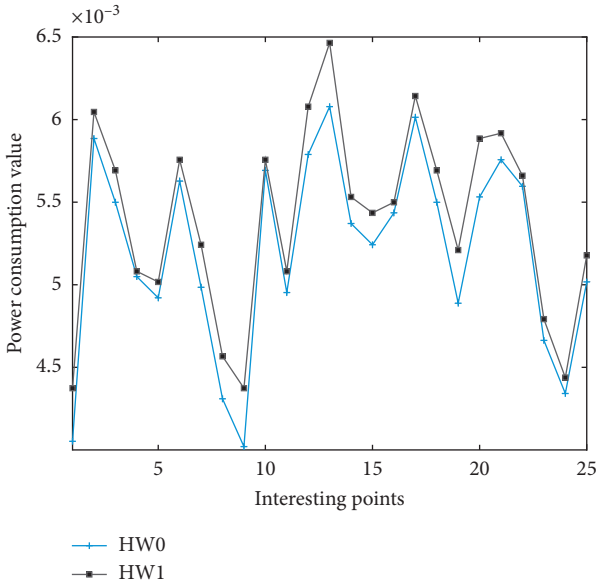


Figure 7: Templates built for serial QUAD.

In the template matching phase, 320 power traces with the same key are captured from the target device. Template that leads to the highest probability indicates the correct key. The success rate of template-based LSQ attack on serial implementation is shown in Figure 8. When the number of power traces approach 30, the success rate tends to 100%. Therefore, a successful attack only requires 30 power traces in serial implementation.

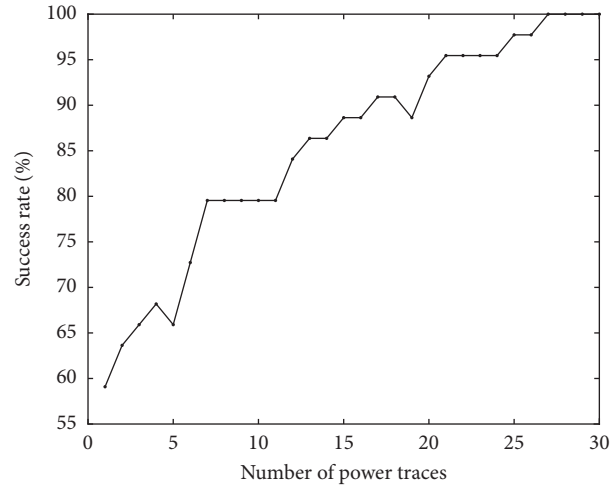


Figure 8: Success rates of our attack on serial implementation.

To further illustrate the effectiveness of our attack, the time required for our practical attack on serial QUAD is discussed here. Our attack is performed on a personal computer, which integrates an Intel i5-7500 CPU and 12 GB of RAM. The time for templates building and templates matching depends on the number of power traces for building and matching, respectively. Figure 9 shows the time required for the template building with the number of power traces ranging from 1 to 3000. Figure 10 shows the time required for template matching with the number of power traces ranging from 1 to 30. As our successful attack on the serial implementation of QUAD (2, 160, 160) requires less than 3000 power traces for templates building and 30 power traces for templates matching, the total time required for our successful attack is less than 1010 seconds, according to Figures 9 and 10.

According to the leakage model of parallel implementation in equation (7), 4 bits of the key are simultaneously accumulated into temporary register. Consequently, we need to guess 4 bits at a time. In the template building phase, 10000 power traces with different keys and coefficients are captured from a reference device, based on which 35 interesting points are selected by the CPA peak method. Next, we collect two groups of power traces corresponding

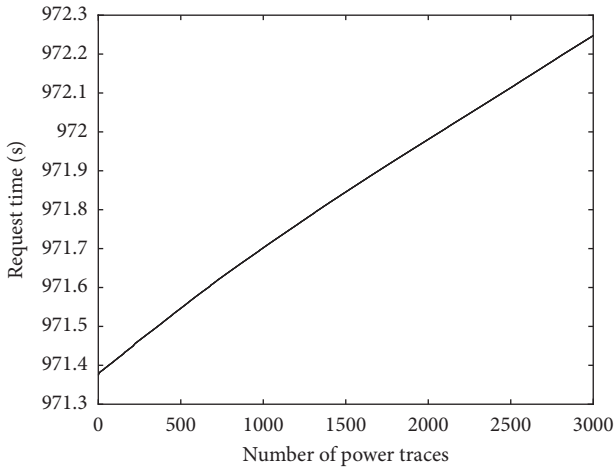


Figure 9: Time required for templates building.

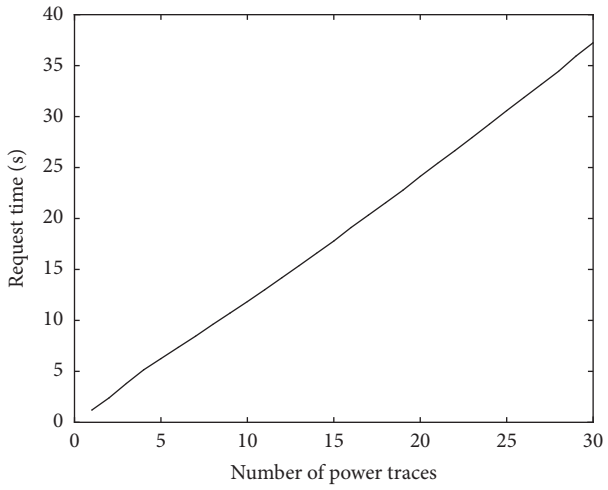


Figure 10: Time required for templates matching of serial QUAD of serial QUAD.

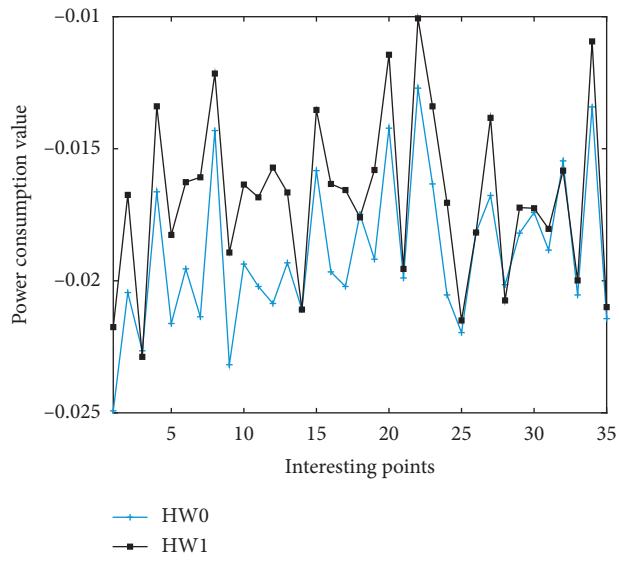


Figure 11: Templates built for parallel QUAD.

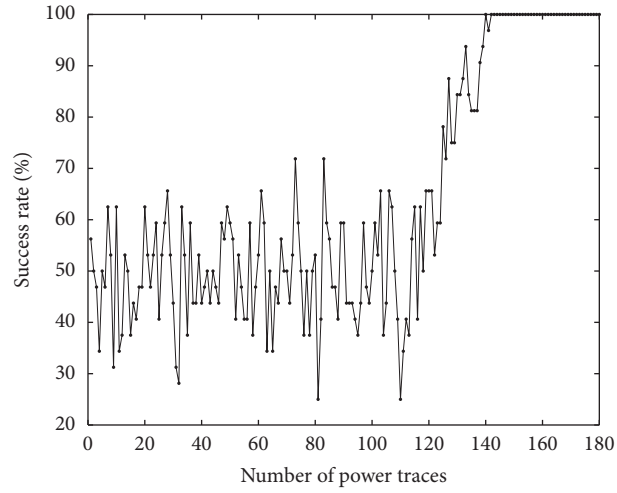


Figure 12: Success rates of our attack on parallel implementation.

to leakage values 0 and 1 in equation (7), and each group consists of 35 power traces. Finally, we build two templates, and the result is shown in Figure 11.

In the template matching phase, 320 power traces with the same key are captured from the target device. Template that leads to the highest probability indicates the correct key. The success rate of template-based LSQ attack on parallel implementation is shown in Figure 12. When the number of power traces approach 150, the success rate tends to 100%. Therefore, 150 power traces are sufficient for a successful attack in parallel implementation.

Figure 13 shows the time required for the template building with the number of power traces ranging from 1 to 10000. Figure 14 shows the time required for template matching with the number of power traces ranging from 1 to 180. As our successful attack on the parallel implementation of QUAD (2, 160, 160) requires less than 10000 power traces for templates building and 150 power traces for templates matching, the total time required for our successful attack is less than 1977 seconds, according to Figures 13 and 14.

In order to compare the success probability of the attacks, we performed our attack, template attack, and template-based DPA attacks dozens of times, respectively. We compare the typical results in Table 1, which shows that our proposed attack has the highest accuracy, greatly outperforming template attack, and template-based DPA attack.

## 5. Suggested Countermeasures

Side-channel countermeasures aim at reducing the data dependency between physical information and secret key. Usually, masking and hiding technologies are adopted. For multivariate cryptography, all monomials and polynomials can be computed in an arbitrary order. Therefore, the basic idea of countermeasures for multivariate cryptography is to randomly change the sequence of these operations.

A QUAD $(q, n, r)$ has $(n + r) \times n(n + 1)/2$ monomials, which can be randomly computed in $((n + r) \times n(n + 1)/2)!$

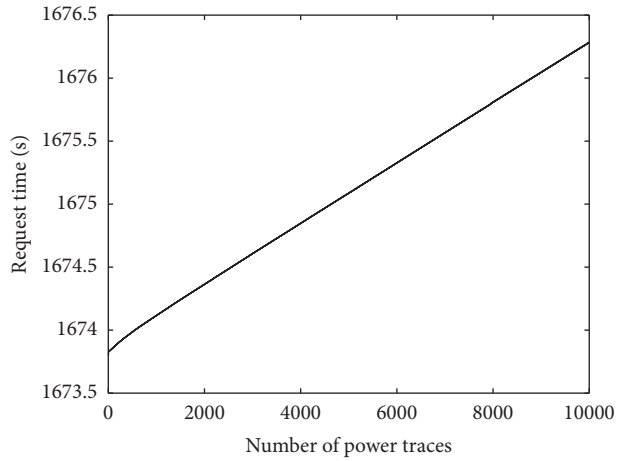


Figure 13: Time required for templates building.

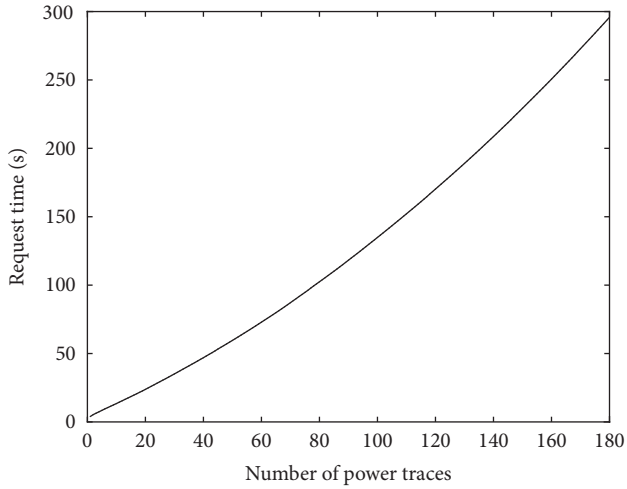


Figure 14: Time required for templates matching of parallel QUAD.

Table 1: Comparison of the success rates of different template attacks.

| No. of experiments/success rate of attack | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Template-based LSQ attack (%) | 98.12 | 97.25 | 100 | 97.25 | 100 | 100 | 100 | 97.25 | 96.875 | 100 |
| Template-based DPA attack (%) | 71.88 | 65.63 | 71.88 | 68.75 | 81.25 | 75 | 81.25 | 78.13 | 68.75 | 75 |
| Template attack (%) | 61.75 | 71.88 | 53.125 | 65.63 | 75 | 68.75 | 61.75 | 65.63 | 63.75 | 59.375 |

orders. However, it is too expensive to implement such algorithm. We propose a low-cost shuffling countermeasure by partially changing the orders of monomials for each polynomial equation $Q_k(X)$. Starting with two randomly generated index $i_s$ and $j_s$, $1 < i_s \le j_s \le n$, each polynomial is computed in the order as follows:

$$Q(x) = \sum_{j_s \le j \le n} \alpha_{i_s j} x_{i_s} x_j + \sum_{(i_s+1) \le i \le j \le n} \alpha_{ij} x_i x_j + \sum_{1 \le i \le (i_s-1), i \le j \le n} \alpha_{ij} x_i x_j + \sum_{i_s \le j \le (j_s-1)} \alpha_{i_s j} x_{i_s} x_j + \gamma. \tag{14}$$

A random index generator is required to generate such an order index, as shown in Figure 15, whose implementation requires only 556 GE.

For the parallel implementations, we proposed a low-cost hiding countermeasure by partially randomizing the initial value of rotated $x$ to shuffle the computation orders of

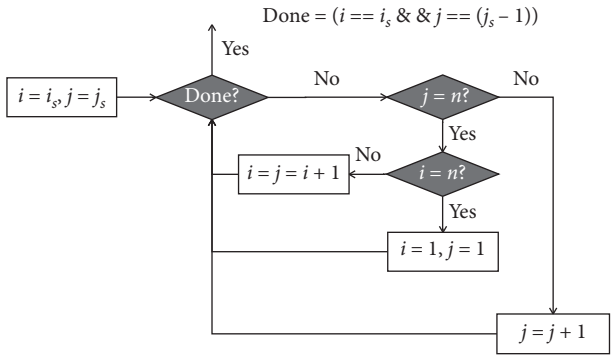


Figure 15: The random index generator of shuffle countermeasure.

monomials. Before each calculation of $Q_k(X)$, the initial value of rotated $x$ is partially randomized with the random starting index $i_s$ as follows:

$$\text{rotated } x = \left(x_{i_s}, x_{i_s+1}, \ldots, x_n, x_1, \ldots x_{i_s-1}\right). \tag{15}$$

## 6. Conclusions

Multivariate cryptosystems consist of a large number of monomials and polynomials, where registers are required to store monomial and polynomial values during the encryption. Therefore, a hamming distance (HD) model of the register will leak the secret of the implementation.

By applying the least-square technique to enable fuzzy matching of the templates, we propose a practical template-based least-square power analysis, where both the serial and parallel implementations of QUAD (2, 160, 160) can achieve a success rate close to 100%. The proposed two low-cost hiding countermeasures for serial and parallel implementations are also validated to be effective, where all monomials and polynomials can be computed in an arbitrary order to break the link between the power consumption and the secret key in multivariate cryptography. Our proposed attacks require only 30 and 150 power traces, respectively, to successfully reveal the secret key. Future work will focus on low-cost countermeasures of multivariate cryptography for IoT devices to resist side-channel attacks.

## Data Availability

The mat data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 61872096, 61672008, and 61772144), Innovation Team Project of the Education Department of Guangdong Province (no. 2017KCXTD021), Guangdong Provincial Key Laboratory of Intellectual Property and Big Data (Grant no. 2018B030322016), Key Laboratory of the Education Department of Guangdong Province (no. 2019KSYS009), and Guangdong Provincial Project of Science and Technology (no. 2016A010101030).

## References

[1] T. Monz, D. Nigg, E. A. Martinez et al., "Realization of a scalable shor algorithm," *Science*, vol. 351, no. 6277, pp. 1068–1070, 2016.

[2] M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang, "Provably secure group signature schemes from code-based assumptions," *IEEE Transactions on Information Theory*, p. 1, 2020.

[3] H. Nejatollahi, N. Dutt, S. Ray, F. Regazzoni, I. Banerjee, and R. Cammarota, "Post-quantum lattice-based cryptography implementations," *ACM Computing Surveys*, vol. 51, no. 6, pp. 1–41, 2019.

[4] T. Takagi, "Recent developments in post-quantum cryptography," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E101.A, no. 1, pp. 3–11, 2018.

[5] D. H. Duong, L. Van Luyen, and H. T. N. Tran, "Choosing subfields for LUOV and lifting fields for Rainbow," *IET Information Security*, vol. 14, no. 2, pp. 196–201, 2020.

[6] J. Ding and D. Schmidt, "Rainbow, a new multivariable polynomial signature scheme," *Applied Cryptography and Network Security*, Springer, New York, NY, USA, pp. 164–175, 2005.

[7] J. Porras, J. Baena, and J. Ding, "ZHFE, a new multivariate public key encryption scheme," in *Proceedings of the International Workshop on Post-Quantum Cryptography*, pp. 229–245, Waterloo, ON, Canada, 2014.

[8] J. Wang, L.-M. Cheng, and T. Su, "Multivariate cryptography based on clipped hopfield neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 2, pp. 353–363, 2018.

[9] C. Berbain, H. Gilbert, and J. Patarin, "QUAD: a practical stream cipher with provable security," *Advances in Cryptology - EUROCRYPT 2006*, in *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques*, pp. 109–128, St. Petersburg, Russia, 2006.

[10] C. Berbain, H. Gilbert, and J. Patarin, "QUAD: a multivariate stream cipher with provable security," *Journal of Symbolic Computation*, vol. 44, no. 12, pp. 1703–1723, 2009.

[11] M. Bardet, J.-C. Faugère, B. Salvy, and P.-J. Spaenlehauer, "On the complexity of solving quadratic boolean systems," *Journal of Complexity*, vol. 29, no. 1, pp. 53–75, 2013.

[12] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, "Secure integration of IoT and cloud computing," *Future Generation Computer Systems*, vol. 78, pp. 964–975, 2018.

[13] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, vol. 107, pp. 841–853, 2017.

[14] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," *Journal of Network and Computer Applications*, vol. 126, pp. 45–58, 2019.

[15] S. Tanaka, C. Cheng, T. Yasuda, and K. Sakurai, "Parallelization of QUAD stream cipher using linear recurring sequences on graphics processing units," in *Proceedings of the 2014 Second International Symposium on Computing and Networking*, pp. 543–548, Shizuoka, Japan, 2014.

[16] G. Liao, Z. Gong, Z. Huang, and W. Qiu, "A generic optimization method of multivariate systems on graphic processing units," *Soft Computing*, vol. 22, no. 23, pp. 7857–7864, 2018.

[17] O. Billet, J. Etrog, and H. Gilbert, "Lightweight privacy preserving authentication for RFID using a stream cipher," *Fast Software Encryption*, in *Proceedings of the 17th International Conference on Fast Software Encryption*, pp. 55–74, Seoul, South Korea, 2010.

[18] D. Arditti, C. Berbain, O. Billet, and H. Gilbert, "Compact FPGA implementations of QUAD," in *Proceedings of the 2nd ACM symposium on Information, computer and communications security-ASIACCS '07*, pp. 347–349, Dallas, TX, USA, 2007.

[19] J. R. Hamlet and R. W. Brocato, "Throughput-optimized implementations of QUAD," *Journal of Cryptographic Engineering*, vol. 5, no. 4, pp. 245–254, 2015.

[20] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Proceedings of the International workshop on Cryptographic Hardware and Embedded Systems (CHES'04)*, pp. 16–29, Cambridge, MA, USA, 2004.

[21] L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F.-X. Standaert, and N. Veyrat-Charvillon, "Mutual information analysis: a comprehensive study," *Journal of Cryptology*, vol. 24, no. 2, pp. 269–291, 2011.

[22] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)*, pp. 13–28, San Francisco Bay (Redwood City), CA, USA, 2002.

[23] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi, "The EM side—channel(s)," in *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)*, pp. 13–28, San Francisco Bay (Redwood City), CA, USA, 2002.

[24] E. Özgen, L. Papachristodoulou, and L. Batina, "Template attacks using classification algorithms," in *Proceedings of the IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 242–247, McLean, VA, USA, 2016.

[25] M. O. Choudary and M. G. Kuhn, "Efficient, portable template Attacks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 2, pp. 490–501, 2018.

[26] L. Lerman, R. Poussier, O. Markowitch, and F.-X. Standaert, "Template attacks versus machine learning revisited and the curse of dimensionality in side-channel analysis: extended version," *Journal of Cryptographic Engineering*, vol. 8, no. 4, pp. 301–313, 2018.

[27] H. Zhang, "On the exact relationship between the success rate of template attack and different parameters," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 681–694, 2019.

[28] S. Picek, A. Heuser, A. Jovic, and L. Batina, "A systematic evaluation of profiling through focused feature selection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 12, pp. 2802–2815, 2019.

[29] B. Hettwer, S. Gehrer, and T. Güneysu, "Applications of machine learning techniques in side-channel attacks: a survey," *Journal of Cryptographic Engineering*, vol. 10, no. 2, pp. 135–162, 2019.

[30] S. Hou, Y. Zhou, H. Liu, and N. Zhu, "Wavelet support vector machine algorithm in power analysis attacks," *Radioengineering*, vol. 26, no. 3, pp. 890–902, 2017.

[31] L. Malina, V. Zeman, J. Martinasek, and Z. Martinasek, "*K*-nearest neighbors algorithm in profiling power analysis attacks," *Radioengineering*, vol. 25, no. 2, pp. 365–382, 2016.

[32] P. Saravanan and P. Kalpana, "A novel approach to attack smartcards using machine learning method," *Journal of Scientific & Industrial Research*, vol. 76, no. 2, p. 99, 2017.

[33] L. Masure, C. Dumas, and E. Prouff, "A comprehensive study of deep learning for side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 1, pp. 348–375, 2020.

[34] G. Zaid, L. Bossuet, A. Habrard, and A. Venelli, "Methodology for efficient CNN architectures in profiling attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 1, pp. 1–36, 2020.

[35] K. Okeya, T. Takagi, and C. Vuillaume, "On the importance of protecting in SFLASH against side channel attacks," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E88-A, no. 1, pp. 123–131, 2005.

[36] Y. Hashimoto, T. Takagi, and K. Sakurai, "General fault attacks on multivariate public key cryptosystems," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E96.A, no. 1, pp. 196–205, 2013.

[37] H. Yi and W. Li, "On the importance of checking multivariate public key cryptography for side-channel attacks: the case of enTTS scheme," *The Computer Journal*, vol. 60, no. 8, pp. 1197–1209, 2017.

[38] A. Park, K. Shim, N. Koo, and D. Han, "Side-channel attacks on post-quantum signature schemes based on multivariate quadratic equations," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 3, pp. 500–523, 2018.

[39] J. Krämer and M. Loiero, "fault attacks on UOV and rainbow," *Constructive Side-Channel Analysis and Secure Design*, in *Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design*, pp. 193–214, Darmstadt, Germany, 2019.

[40] W. Li, F. Lu, and H. Zhao, "Power analysis attacks against QUAD," *IAENG International Journal of Computer Science*, vol. 46, no. 1, pp. 54–60, 2019.